

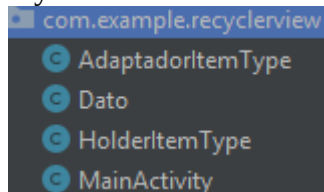
RecyclerViewPropuesto

Primero creamos el xml con el anunciado correspondiente. Y debemos crearnos un recyclerView con la dependencia correspondiente. En la actividad principal de xml al que le vamos a llamar con principal clase de java.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recycler"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Y creamos 3 clases más en nuestro proyecto:



- com.example.recyclerview
 - AdaptadorItemType
 - Dato
 - HolderItemType
 - MainActivity

En Holder extendemos con la clase ViewHolder y implementamos OnClickListener: y creamos

```
public class HolderItemType extends RecyclerView.ViewHolder implements View.OnClickListener {

    Context context;
    View itemView;
    ImageView itemTypeImage;
    TextView itemTypeTextView;
    ImageView favorito;
    ImageView turned_in;
    ImageView social;
    Button bShare;
    Button bExplore;
    MainActivity.OnImagenClickListener listener;
    Dato dato;
```

Buscamos los objetos relacionando con el xml que hemos creado. Que en el constructor le pasamos la view, y la context, y el tipo indicado a cada objeto.

Con un switch manejo la view de un xml indicado.

```

public HolderItemType(View itemView, Context context, int tipo) {
    super(itemView);
    this.context = context;
    this.itemView = itemView;

    itemTypeImage = itemView.findViewById(R.id.imageView);
    itemTypeTextView = itemView.findViewById(R.id.txt_title);
    itemTypeImage.setOnClickListener(this);

    switch (tipo) {
        case 0:
            break;
        case 1:
            favorito = itemView.findViewById(R.id.imageView1);
            turned_in = itemView.findViewById(R.id.imageView2);
            social = itemView.findViewById(R.id.imageView3);

            favorito.setOnClickListener(this);
            turned_in.setOnClickListener(this);
            social.setOnClickListener(this);
            break;
        case 2:
            bShare = itemView.findViewById(R.id.bShare);
            bExplore = itemView.findViewById(R.id.bExplore);

            bShare.setOnClickListener(this);
            bExplore.setOnClickListener(this);
            break;
    }
}

```

En el método bind le paso el dato y coloco la foto y el texto que viene con el dato.

```

public void bind(Dato dato, int pos){
    //itemView.setBackgroundColor(ContextCompat.getColor(context,R.color.colorAccent))
    itemTypeImage.setImageBitmap(dato.getFoto());
    itemTypeTextView.setText(dato.getTextoCorto());
    this.dato = dato;
}

```

Al final de la clase le asigno Click Listener que tengo creado en MainActivity una interfaz:

```

public void setClickAImagenListener(MainActivity.OnImagenClickListener listener){
    if(listener!=null)
        this.listener = listener;
}

@Override
public void onClick(View view) {
    if (listener != null) listener.OnImageClick(dato, view);
}

```

Esta es la interfaz del activity principal:

```
public interface OnImagenClickListener {  
    void OnImageClick(Dato d, View v);  
}
```

Claro necesitamos un adaptador para controlar el Holder:

le extiendo con RecyclerView Adapter y implemento on click listener:

Creo los objeto que necesito para la clase:

al constructor le paso el contexto principal: de la MainActivity:

y Sobre Escribo un método getItemViewType(int pos) y hay me cargo los datos con la posición que me da el recycler view. Y casteo a dato en Datos

y devuelvo su tipo de layout que vamos a utlizar en el siguiente método:

```
public class AdaptadorItemType extends RecyclerView.Adapter implements View.OnClickListener {  
  
    Context c;  
    HolderItemType h;  
    MainActivity.OnImagenClickListener listener;  
  
    Dato d;  
  
    public AdaptadorItemType(Context c) {  
        this.c = c;  
    }  
  
    @Override  
    public int getItemViewType(int position) {  
        d = (Dato) (((MainActivity)c).datosLista).get(position);  
        return d.getTipo();  
    }  
}
```

Este Métopd también se sobre escribe pero hay que devolver un holder. Pues el getItemViewType lo que devuelve es el “viewType” de este método. primero sacamos el view Gracias a Layoutflater lo sacamos con el viewType correspondiente.

Y pasamos al holder el view, el contexto y el tipo de la vista.

Al holder le ponemos un setClickListener para que nos pile los clicks sobre el.

Y creamos el listener de nuestra interfaz por donde tiene que dirigir. Con su Vista y dato correspondiente.

Y al final retornamos el holder.

```

@Override
public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {

    View view = null;
    System.out.println(getItemViewType(viewType) + "HOLA");

    switch (viewType) {
        case 0:
            view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_type_1, parent, attachToRoot: false);
            break;
        case 1:
            view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_type_2, parent, attachToRoot: false);
            break;
        case 2:
            view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_type_3, parent, attachToRoot: false);
            break;
        default:
            break;
    }

    h = new HolderItemType(view, c, viewType);
    view.setOnClickListener(this);

    h.setClickAIMagenListener(new MainActivity.OnImagenClickListener() {
        @Override
        public void OnImageClick(Dato d, View v) {
            listener.OnImageClick(d, v);
        }
    });

    return h;
}

```

Al final en onBindViewHolder llamamos el bind del holder creado (para que nos ponga la imagen y el texto de cada tarjeta)

en getItemCount() solamente hay que retornar la longitud de datos correspondiente.

El onClick no lo utilizo porque tengo mi interfaz y aprovecho y lo creo para crear el onClick

```

@Override
public void onBindViewHolder(RecyclerView.ViewHolder holder, int position) {

    ((HolderItemType)holder).bind(d, position);
}

@Override
public int getItemCount() {
    return ((MainActivity)c).datosLista.size();
}

public void setClickAIMagenListener(MainActivity.OnImagenClickListener listener){
    if(listener!=null)
        this.listener = listener;
}

@Override
public void onClick(View v) {
}

```

En Actividad principal creo los objetos necesarios.

```
public class MainActivity extends AppCompatActivity {

    protected ArrayList<Dato> datosLista = new ArrayList<>();
    private static final int TYPE_ITEM_1 = 0;
    private static final int TYPE_ITEM_2 = 1;
    private static final int TYPE_ITEM_3 = 2;

    RecyclerView recyclerView;
    AdaptadorItemTipo adaptadorItemTipo1;
}
```

En OnCreate Relleno los datos:
 Busco el recyclerView de mi actividadPrincipal de xml.
 Creo el adaptador y le paso el contexto actual.
 Le pongo un tamaño fijo.
 Y el layout le pongo como vertical.

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    final View view = findViewById(R.id.activity_main);

    datosLista.add(new Dato( textoCorto: "Cohete espacial", textolargo: "Soy un cohete espacial que viaja por el espacio interestelar",BitmapFactory.decodeResource(this.getResources(), R.drawable.cohete), BitmapFactory.decodeResource(this.getResources(), R.drawable.cohete)));
    datosLista.add(new Dato( textoCorto: "Cordillera de noche", textolargo: "No hay nada como una noche plácida en la montaña, ¿verdad?",BitmapFactory.decodeResource(this.getResources(), R.drawable.cordillera), BitmapFactory.decodeResource(this.getResources(), R.drawable.cordillera)));
    datosLista.add(new Dato( textoCorto: "London city", textolargo: "No hay nada como pasear por la grilla del Támesis en una mañana con niebla",BitmapFactory.decodeResource(this.getResources(), R.drawable.london), BitmapFactory.decodeResource(this.getResources(), R.drawable.london)));
    datosLista.add(new Dato( textoCorto: "Discovery en la noche", textolargo: "Viajar al espacio, recorrer la vía láctea, y volver a casa con mi Discovery...", BitmapFactory.decodeResource(this.getResources(), R.drawable.discovery), BitmapFactory.decodeResource(this.getResources(), R.drawable.discovery)));

    recyclerView = findViewById(R.id.recycler);
    adaptadorItemTipo1 = new AdaptadorItemTipo( this );
    recyclerView.setAdapter(adaptadorItemTipo1);
    recyclerView.setHasFixedSize(true);
    recyclerView.setLayoutManager(new LinearLayoutManager( context: this, LinearLayoutManager.VERTICAL, reverseLayout: false));

    adaptadorItemTipo1.setClickListener(new OnItemClickListener() {
        @Override
        public void onItemClick(Dato d, View v) {
            switch (v.getId()) {
                case R.id.imageView:
                    Toast.makeText( context: MainActivity.this, d.getTextoLargo(), Toast.LENGTH_LONG).show();
                    break;
                case R.id.imageView1:
                    Toast.makeText( context: MainActivity.this, text: "Favorito", Toast.LENGTH_LONG).show();
                    break;
                case R.id.imageView2:
                    Toast.makeText( context: MainActivity.this, text: "Social", Toast.LENGTH_LONG).show();
                    break;
                case R.id.imageView3:
                    Toast.makeText( context: MainActivity.this, text: "Turned in", Toast.LENGTH_LONG).show();
                    break;
                case R.id.bExplore:
                    Toast.makeText( context: MainActivity.this, text: "Button Explore", Toast.LENGTH_LONG).show();
                    break;
                case R.id.bShare:
                    Toast.makeText( context: MainActivity.this, text: "Button share", Toast.LENGTH_LONG).show();
                    break;
            }
        }
    });

    recyclerView.setAdapter(adaptadorItemTipo1);
}
```

La interfaz que creamos para el click utilizamos aquí obteniendo el dato y la vista del mismo objeto.
 Con un switch veo los identificadores de nuestros xml
 Y al final a recyclerView hay que ajustar el adaptador.

```
adaptadorItemTipo1.setClickListener(new OnItemClickListener() {
    @Override
    public void onItemClick(Dato d, View v) {
        switch (v.getId()) {
            case R.id.imageView:
                Toast.makeText( context: MainActivity.this, d.getTextoLargo(), Toast.LENGTH_LONG).show();
                break;
            case R.id.imageView1:
                Toast.makeText( context: MainActivity.this, text: "Favorito", Toast.LENGTH_LONG).show();
                break;
            case R.id.imageView2:
                Toast.makeText( context: MainActivity.this, text: "Social", Toast.LENGTH_LONG).show();
                break;
            case R.id.imageView3:
                Toast.makeText( context: MainActivity.this, text: "Turned in", Toast.LENGTH_LONG).show();
                break;
            case R.id.bExplore:
                Toast.makeText( context: MainActivity.this, text: "Button Explore", Toast.LENGTH_LONG).show();
                break;
            case R.id.bShare:
                Toast.makeText( context: MainActivity.this, text: "Button share", Toast.LENGTH_LONG).show();
                break;
        }
    }
});

recyclerView.setAdapter(adaptadorItemTipo1);
}
```

Y la clase Dato se crea muy facil:

```

class Dato {

    private String textoCorto;
    private String textoLargo;
    private Bitmap foto;
    private int tipo;

    public Dato(String textoCorto, String textoLargo, Bitmap foto, int tipo) {
        this.textoCorto = textoCorto;
        this.textoLargo = textoLargo;
        this.foto = foto;
        this.tipo = tipo;
    }

    public String getTextoCorto() {
        return textoCorto;
    }

    public void setTextoCorto(String textoCorto) {
        this.textoCorto = textoCorto;
    }

    public String getTextoLargo() {
        return textoLargo;
    }

    public void setTextoLargo(String textoLargo) {
        this.textoLargo = textoLargo;
    }
}

```

```

    public Bitmap getFoto() {
        return foto;
    }

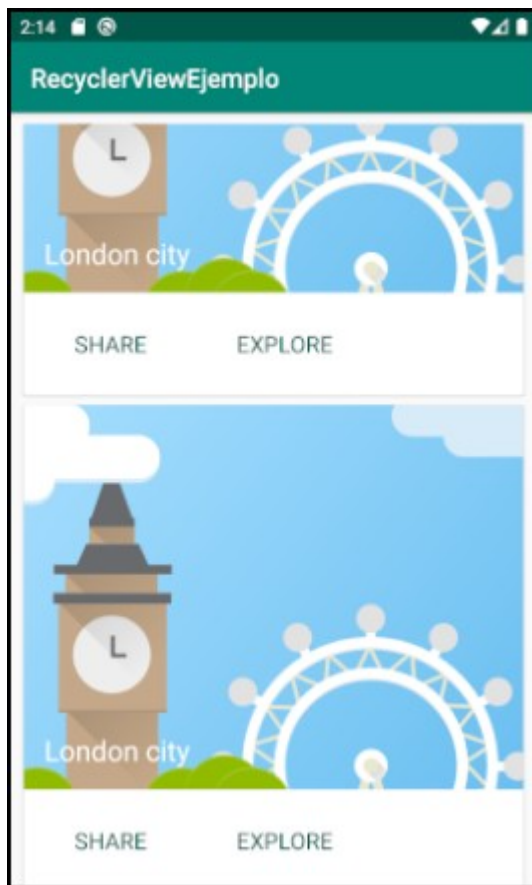
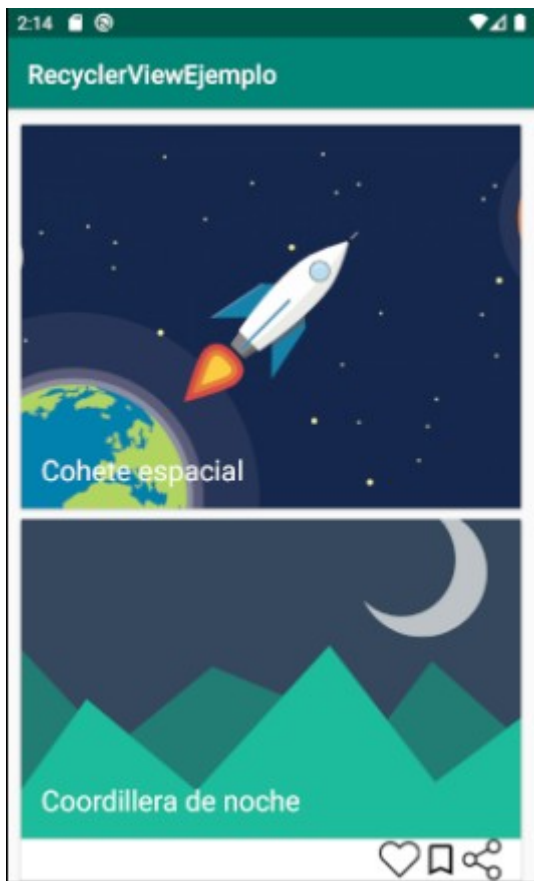
    public void setFoto(Bitmap foto) {
        this.foto = foto;
    }

    public int getTipo() {
        return tipo;
    }

    public void setTipo(int tipo) {
        this.tipo = tipo;
    }
}

```

Y ya esta, ya hemos terminado.



Y cuando pulsamos en cada mImagen o en cada boton aparece el mensaje del este boton o el dato desde cada objeto Dato.